SANDIA REPORT

SAND2016-10753
Unlimited Release
Printed June 2012

ML Enhancements via the Calculation of Rigid Body Modes (RBMs) for Mechanics Problems Implemented within the Albany Code Base

Irina Kalashnikova

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2016-10753
Unlimited Release
Printed June 2012

ML Enhancements via the Calculation of Rigid Body Modes (RBMs) for Mechanics Problems Implemented within the Albany Code Base

Irina Kalashnikova
Numerical Analysis and Applications Department
Sandia National Laboratories
P.O. Box 5800, MS 1320
Albuquerque, NM 87185-1320
ikalash@sandia.gov

Abstract

The algebraic multigrid approach known as smoothed aggregation is very efficient at solving systems that arise from elasticity problems [1]. In order to construct an efficient algebraic multilevel method, a multigrid solver should be provided with a small set of vectors that represent the error components that are difficult to resolve. It is well-known [2, 5] that for linear elasticity problems, these components correspond to the the so-called rigid body modes (RBMs). The present document summarizes some new development within the Albany code base that has enabled the application of algebraic multigrid preconditioners from the `ML` package [2] of Trilinos to mechanics problems implemented within Albany via a new function that calculates the RBMs using information about the problem's underlying mesh. The performance of these preconditioners is evaluated on four problems: a 3D static elasticity problem, a 3D non-linear elasticity problem, a 3D thermo-elasticity problem, and a 3D thermo-poro-plasticity problem. The tests reveal the superiority of the `ML` preconditioners over ILU preconditioners from the Trilinos `Ifpack` package [4] for mechanics problems in Albany.

Contents

Figures

Tables

1 Introduction

The basic idea of a multigrid solver is to accelerate the convergence of an iterative method applied to a problem arising from the discretization of a partial differential equation (PDE) by solving the problem on a hierarchy of "coarse" grids, and interpolating between these "coarse" grids and the original "fine" grid. A popular efficient multigrid approach – one that is implemented within the `ML` package of Trilinos [2] – is smoothed aggregation. In smoothed aggregation, a grid transfer operator is constructed from a seed vector (or set of seed vectors), following the aggregation of the vertices of a matrix graph (coarsening). This aggregation (or coarsening) and seed vector information is used to create a tentative interpolation operator, one that by construction will interpolate the seed vectors perfectly. Following aggregation, smoothers are applied on each level to damp out errors in the solution. It is well-known [2] that errors in the direction of eigenvectors of the discretization operator corresponding to small eigenvalues are difficult to smooth. Hence, a common strategy is to select as the seed vectors the vectors in the nullspace or near-nullspace of a problem's discretization matrix.

For some classes of problems, the constant vectors used as seed vectors by `ML` by default are *not* necessarily in the nullspace of the problem operator, and therefore not the best choices for the seed vectors used in smoothed aggregation. A classical example of such a problem is elasticity. For elasticity, it is well-known that the nullspace of the operator is spanned by a floating structure's rigid body modes (RBMs). These modes are in effect the directions in which a body that is not adequately supported can translate or rotate as a whole without deformation. For three-dimensional (3D) elasticity, a floating structure has six RBMs (three translational and three rotational vectors); for two-dimensional (2D) elasticity, a floating structure has three RBMs (two translational and one rotational vectors); for one-dimensional (1D) elasticity, a floating structure has just one (translational) RBM. To ensure good performance of an `ML` preconditioned iterative method applied to solve discretized elasticity equations, it is important that the RBMs be explicitly provided to the multigrid solver (`ML`).

The present document summarizes the results of a numerical study that evaluates the relative performance of `Ifpack` and `ML` preconditioners for mechanics problems implemented within the Albany code base. To enable proper computation of seed vectors for `ML`, the following function was developed within Albany:

```
Albany_ML_Coord2RBM(int Nnodes, double x[], double y[], double z[], double
rbm[], int Ndof, int NscalarDof, int NSdim)
```

The remainder of this document is organized as follows. Section 2 describes the implementation of the `Albany_ML_Coord2RBM` function for aiding the construction of `ML` preconditioners for mechanics problems implemented within Albany. The performance of these preconditioners is evaluated in Section 3 on four mechanics problems: a 3D static elasticity problem, a 3D non-linear elasticity problem, a 3D thermo-elasticity problem, and a 3D thermo-poro-plasticity problem. Performance comparisons are made to ILU preconditioners from the `Ifpack` package. Conclusions are offered in Section 4.

2 Calculation of Rigid Body Modes (RBMs) for Mechanics Problems in Albany

The Albany classes have been designed to communicate automatically the correct seed vector information to the multigrid solver `ML` for each problem implemented within the Albany code base. The virtual function:

```
getRBMInfoForML(int numPDEs, int numElasticityDim, int numScalar,
int nullSpaceDim)
```

has been added to the `Albany::AbstractProblem` class (`src/problems/Albany_AbstractProblem.cpp`). The number of elasticity equations (`numElasticityDim`) is set to zero by default. If an `ML` preconditioner is selected in the Albany `.xml` input file and `numElasticityDim` is set to zero, the `ML` preconditioner will be created using the default constant seed vectors. For elasticity problems implemented with the Albany Laboratory for Computational Mechanics (LCM) suite (`src/LCM/problems`), `numElasticityDim` is overwritten from its default value of zero. This integer parameter is set to the number of elasticity equations being solved (1 for 1D problems, 2 for 2D problems, and 3 for 3D problems). For some problems within the LCM suite, the elasticity equations are coupled to one or more PDEs. The integer `numScalar` specifies the number of PDEs coupled to elasticity for a given problem. For instance, for the thermo-elasticity problem, the elasticity equations are coupled to a single PDE, namely the heat equation, so `numScalar = 1`. For the thermo-poro-plasticity problem, which couples balance of linear momentum, mass and energy (used to model solid deformation) to fluid and thermal diffusion, `numScalar = 2`.

Table 1. Inputs and outputs to `Albany_ML_Coord2RBM` function

	Name	Description	Details
Inputs	`Nnodes`	# nodes in physical mesh	–
	`x`, `y`, `z`	arrays holding nodal coordinates	–
	`Ndof`	total # PDE DOFs per node	= # elasticity DOFs per node +`NscalarDof`
	`NscalarDof`	# scalar (non-elasticity) DOFs per node	–
	`NSdim`	dimension of elasticity nullspace	$= \begin{cases} 1, & \text{for 1D elasticity} \\ 3, & \text{for 2D elasticity} \\ 6, & \text{for 3D elasticity} \end{cases}$
Outputs	`rbm`	array populated with RBMs	size: `Nnodes*Ndof*(NSdim + NscalarDof)`

For elasticity problems (`numElasticityDim` $\neq$ `0`) solved with an `ML` preconditioned iterative method, the `ML` seed vectors are computed via the following newly-developed function:

```
Albany_ML_Coord2RBM(int Nnodes, double x[], double y[], double z[], double
rbm[], int Ndof, int NscalarDof, int NSdim)
```

Table 1 summarizes the input and output parameters to this function. The scalar input parameters in Table 1 are generated for each problem implemented in the Albany code base by an aforementioned function `getRBMInfoForML`. The vector input parameter, namely the arrays containing the x–, y– and z–coordinates of the mesh are obtained through another new Albany function, `setCoordinatesForML`, implemented in `Albany_SolverFactory.cpp`.

Illustrative example of RBM calculation:

As an example, the seven $\mathbb{R}^7$ nullspace vectors corresponding to each coordinate for a 3D elasticity problem discretized with a shell element coupled to a scalar PDEs is demonstrated in Table 2.

Table 2. Nullspace vectors for 3D elasticity shell problem coupled to a scalar PDE

	translations (elasticity)			scalar	rotations around (elasticity)		
	x	y	z	eq. #1	x	y	z
x–direction (elasticity)	1	0	0	0	0	z	$-y$
y–direction (elasticity)	0	1	0	0	$-z$	0	x
z–direction (elasticity)	0	0	1	0	y	$-x$	0
scalar eq. #1	0	0	0	1	0	0	0
x–rotation (elasticity)	0	0	0	0	1	0	0
y–rotation (elasticity)	0	0	0	0	0	1	0
z–rotation (elasticity)	0	0	0	0	0	0	1

The nullspace vectors for 3D elasticity with bricks would be the same as Table 2 but with the last three rows removed. The nullspace vectors for 2D elasticity would be generated by removing also remove the third row and columns three, five and six from Table 2. As expected, generating the nullspace vectors for 1D elasticity would require removing all but rows one and four and columns one and four in the table.

3 Numerical Results

This section summarizes the results of a study aimed at evaluating the relative performance of different preconditioners available through the Trilinos `Ifpack` and `ML` packages for various mechanics problems implemented within the Albany LCM suite. Seven basic preconditioner types are considered: four `Ifpack` preconditioners and three `ML` preconditioners (Table 3). The `Ifpack` preconditioners are effectively ILU preconditioners, and differ in the `overlap` and `level-of-fill` options. The `ML` preconditioners are algebraid multi-grid preconditioners based on three default preconditioner types available in the `ML` package: `SA` (classical Smoothed Aggregation), `DD` (classical smoothed aggregation based on two-level Domain Decomposition), and `DD-ML` (three-level algebraic Domain Decomposition). For a detailed discussion of these `Ifpack` and `ML` options, the reader is referred to the `Ifpack` and `ML` user guides, [4] and [2] respectively. The `ML` preconditioners all employ the matrix repartitioning option available through the Trilinos `Zoltan` package. Essentially, repartitioning uses information about the mesh coordinates to perform dynamic load-balancing of coarse-level matrices in the multigrid preconditioner. With repartitioning, message passing latency on the coarse level can be improved, and the well-known problem of the coarsening rate dropping as the number of unknowns per processor becomes small can be avoided.

Table 3. Summary of preconditioners evaluated

Preconditioner #	Type	Parameters
1	ifpack	`overlap = 1, level-of-fill = 1`
2		`overlap = 2, level-of-fill = 1`
3		`overlap = 1, level-of-fill = 2`
4		`overlap = 2, level-of-fill = 2`
5	ML	`default type = SA`
6		`default type = DD`
7		`default type = DD-ML`

The following subsections summarize the performance results of the `ML` preconditioners for four benchmark problem: a 3D static elasticity problem, a 3D non-linear elasticity problem, a 3D thermo-elasticity problem, and a 3D thermo-poro-plasticity problem. Three mesh resolutions h are considered. For the first three problems, the mesh resolutions are: $h = 50$ ($50 \times 50 \times 50$ element mesh), $h = 100$ ($100 \times 100 \times 100$ element mesh), $h = 200$ ($200 \times 200 \times 200$ element mesh). For the final problem, the mesh resolutions are: $h = 25$ ($25 \times 25 \times 25$ element mesh), $h = 50$ ($50 \times 50 \times 50$ element mesh), $h = 75$ ($75 \times 75 \times 75$ element mesh). All meshes are comprised of 3D brick elements generated using the Sierra Toolkit (STK). For the thermo-elasticity problem, the governing equations are the elasticity equations coupled to a scalar head equation (`numScalar = 1`). For the thermo-poro-plasticity problem, equations for the balance of linear momentum, mass and energy (used to model solid deformation) are coupled to equations of fluid and thermal diffusion (`numScalar = 2`). For all problems, `Belos` total linear solve and total preconditioner creation times[1] for each preconditioner considered are reported for each problem and all three

[1] Included in the total linear solve time.

mesh sizes considered. The iterative method employed for the linear solves was preconditioned GMRES.

3.1 3D Static Elasticity

The 3D static elasticity problem considered here is both stationary and linear. Figure 1 reports the total linear solve and total preconditioner creation times for the smallest static elasticity problem considered, solved on a $50 \times 50 \times 50$ STK mesh. For this mesh resolution, the linear solver converged and the correct solution was obtained with all seven preconditioners considered. The reader may observe by examining Figure 1 that the linear solves are at least five times faster with an `ML` versus an `Ifpack` preconditioner. Figure 2 reports the total linear solve and total precon-

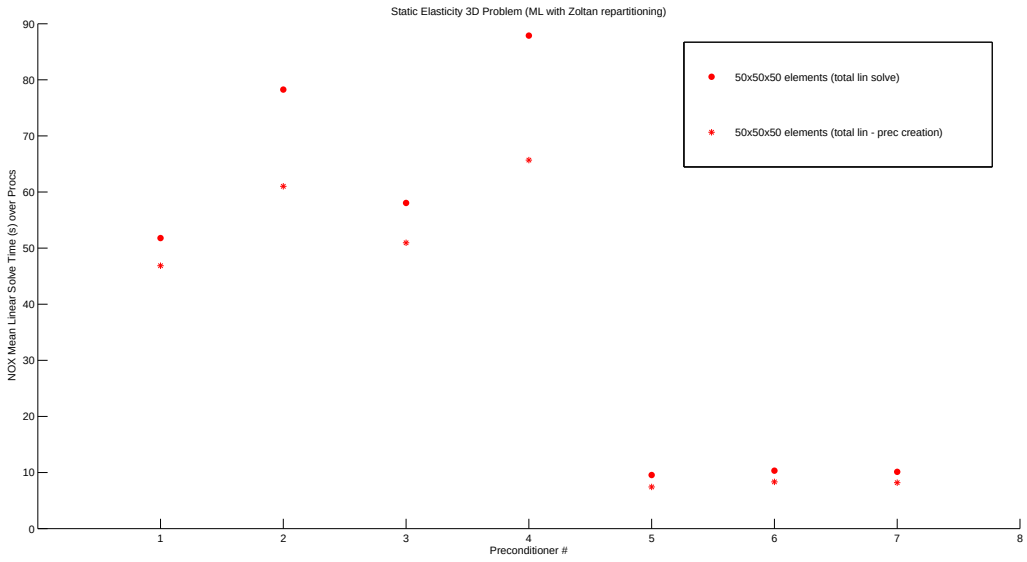

Figure 1. 3D Static Elasticity: `Ifpack` vs. `ML` preconditioners with `Zoltan` repartitioning

ditioner creation times for the static elasticity problem on all three mesh resolutions considered. Results in this figure are not shown for the `Ifpack` preconditioners because, with this choice of preconditioner, the linear solver failed to converge and an incorrect solution was obtained.

3.2 3D Non-linear Elasticity

Figure 3 reports the total linear solve and total preconditioner creation times for the non-linear elasticity problem on all three mesh resolutions considered. As for the static elasticity problem posed on the finer meshes, the linear solver fails to converge with the choice of an `Ifpack` preconditioner. Consequently, the non-linear solver fails to converge, and an incorrect solution is obtained.

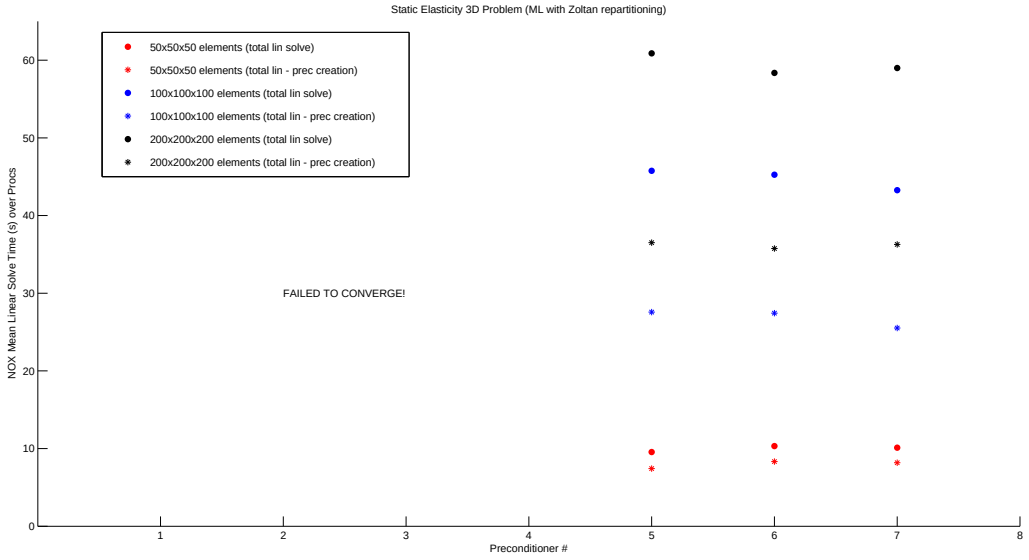

Figure 2. 3D Static Elasticity: `ML` preconditioners with `Zoltan` repartitioning

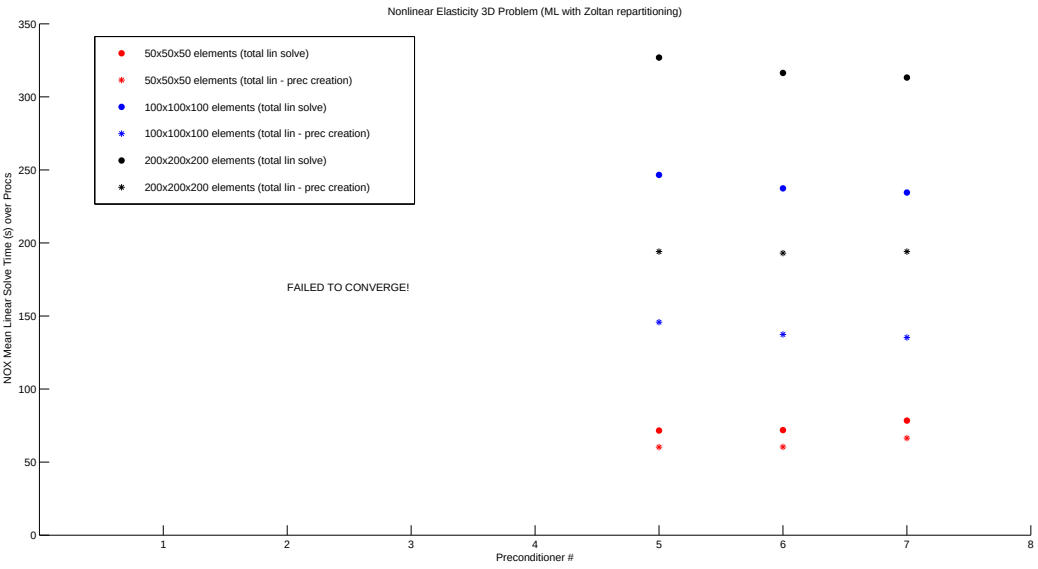

Figure 3. 3D Non-linear Elasticity: `ML` preconditioners with `Zoltan` repartitioning

3.3 3D Thermo-elasticity

The third problem considered is the so-called thermo-elasticity problem. The equations being solved are 3D elasticity equations coupled to a scalar heat equation. As for the static elasticity

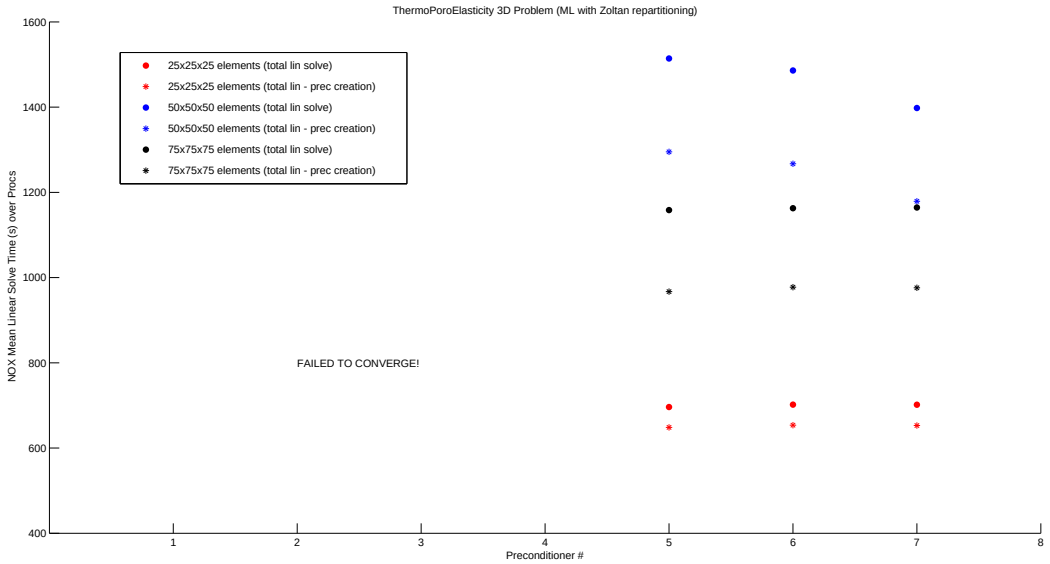

Figure 4. 3D Thermo-elasticity: `Ifpack` vs. `ML` preconditioners with `Zoltan` repartitioning

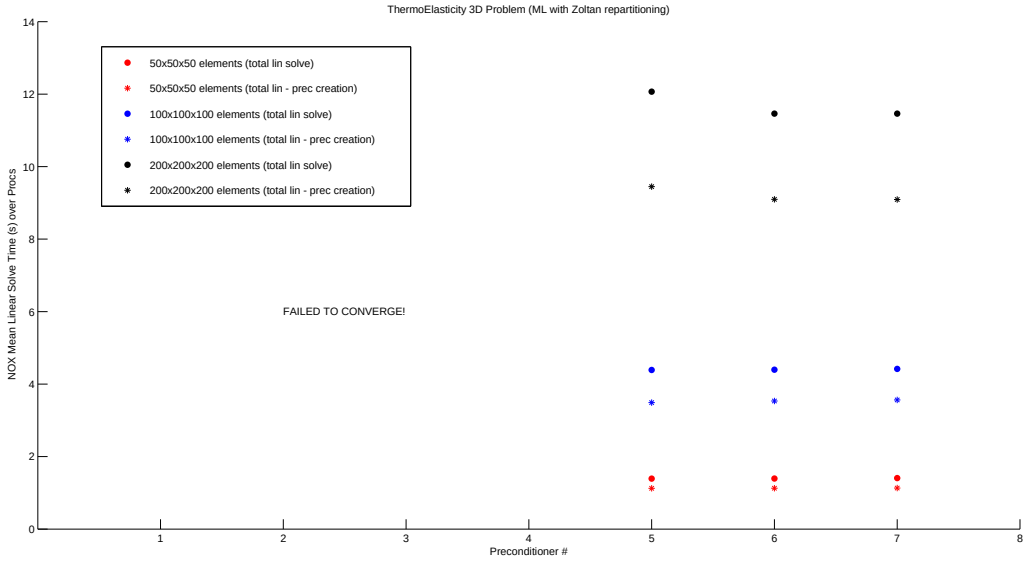

Figure 5. 3D Thermo-elasticity: `ML` preconditioners with `Zoltan` repartitioning

problem, the solvers converged with the choice of an `Ifpack` preconditioner on the coarsest mesh considered (Figure 4). For this mesh resolution, there is only a slight decrease in total linear solve time with the choice of an `ML` preconditioner over the `Ifpack` preconditioner with `level-of-fill` and `overlap` both set to 1. The `ML` preconditioners are nonetheless recommended, as the linear

solver fails to converge with an `Ifpack` preconditioner on the two finer mesh resolutions (Figure 5).

3.4 3D Thermo-poro-plasticity

The final problem considered is the 3D thermo-poro-plasticity problem. For this problem, equations for the balance of linear momentum, mass and energy (to simulate solid deformation) are coupled to equations for fluid and thermal diffusion. Coupling terms are then added to the three residual equations (linear momentum, mass and energy) so that the fully coupled system can be solved monolithically. For more information on the details of the thermo-poro-plasticity problem, the reader is referred to [3]. Total linear solve times and preconditioner creation times for this problem are reported in Figure 6. The reader can observe that the non-linear solver fails to converge if an `Ifpack` preconditioner is employed to solve the linear systems arising within each non-linear step. In contrast, the solver converges to the correct solution if the linear systems are preconditioned with an `ML` preconditioner.

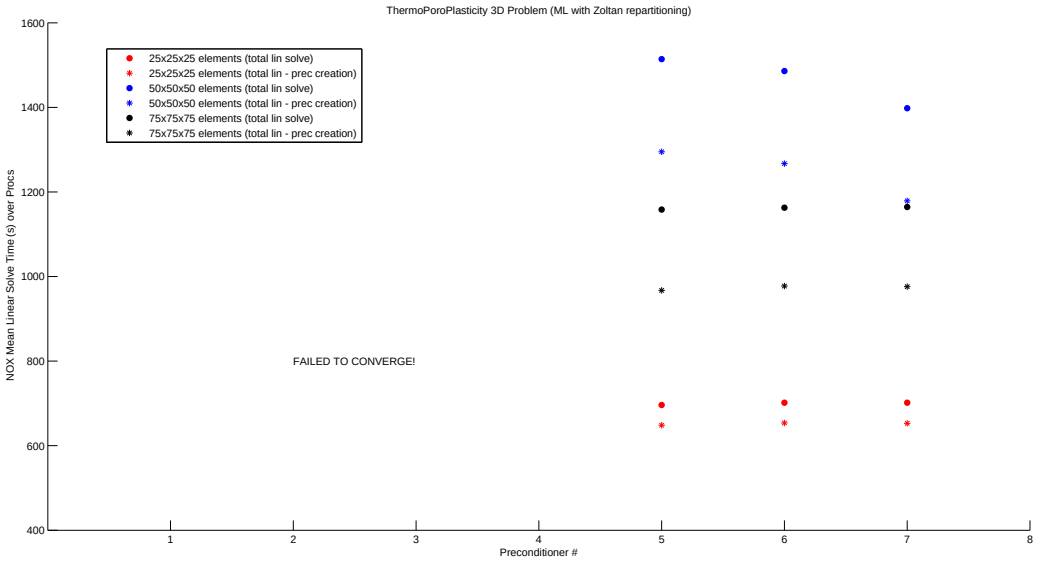

Figure 6. 3D Thermo-poro-plasticity: `ML` preconditioners with `Zoltan` repartitioning

4 Conclusions

A new capability that has enabled the application of algebraic multigrid preconditioners from the `ML` package [2] of Trilinos to mechanics problems implemented within the Albany code base is described. The new development within Albany focuses around the efficient implementation of a function that calculates the elasticity rigid body modes (RBMs) using information about the problem's underlying mesh. These modes represent the error components that are difficult for a multigrid solver applied to elasticity problems to resolve. The RBM calculation function is extended to provide the multigrid solver with the proper set of seed vectors for the more complicated case when the problem of interest involves elasticity equations coupled to one or more additional PDEs. The performance of the `ML` preconditioners constructed using these seed vectors is evaluated on four problems within the Albany Laboratory for Computational Mechanics (LCM) suite. Whereas the ILU preconditioners from the Trilinos `Ifpack` package are in general inadequate, especially for larger problems discretized by finer meshes, convergence of the `ML`-preconditioned linear solves, and subsequently the non-linear solves, is observed for all test cases.

References

[1] M. Brezina, R. Falgout, S. Maclachlan, T. Manteuffel, S. McCormick, and J. Ruge. Adaptive smoothed aggregation (αsa). *J. Sci. Comput.*, 25(6):1896–1920, 2004.

[2] M.W. Gee, C.M. Siefert, J.J. Hu, R.S. Tuminaro, and M.G. Sala. Ml 5.0 smoothed aggregation user's guide. Technical report, Sandia National Laboratories, 2007.

[3] M. Preisig and J.H. Prevost. Coupled multi-phase thermo-poromechanical effects. case study: Co_2 injection at in salah, algeria. *J. Greenhouse Gas Control*, 5:1055–1064, 2011.

[4] M. Sala and M. Heroux. Robust algebraic preconditioners using ifpack 3.0. Technical report, Sandia National Laboratories, 2005.

[5] P. Vanek, M. Brezina, and J. Mandel. Convergence of algebraic multigrid based on smoothed aggregation. Technical report, UCD/CCM, 1998.

DISTRIBUTION:

1	MS 1318	Andy Salinger, 1442
1	MS 9159	Ray Tuminaro, 1442
1	MS 1322	Chris Seifert, 1443
1	MS 1320	Scott Collis, 1442
1	MS 9159	Jonathan Hu, 1426
1	MS 9042	Jake Ostien, 8256
1	MS 9912	WaiChing Sun, 8256
1	MS 1320	Scott Collis, 1442
1	MS 0899	Technical Library, 9536 (electronic copy)

Sandia National Laboratories